MICROCONTROLLER AND EMBEDDED SYSTEMS

SEMESTER 5 IT ENGINEERING(MU) - COMPLETE COVERAGE

OVERVIEW OF EMBEDDED SYSTEMS

BASIC CONCEPTS

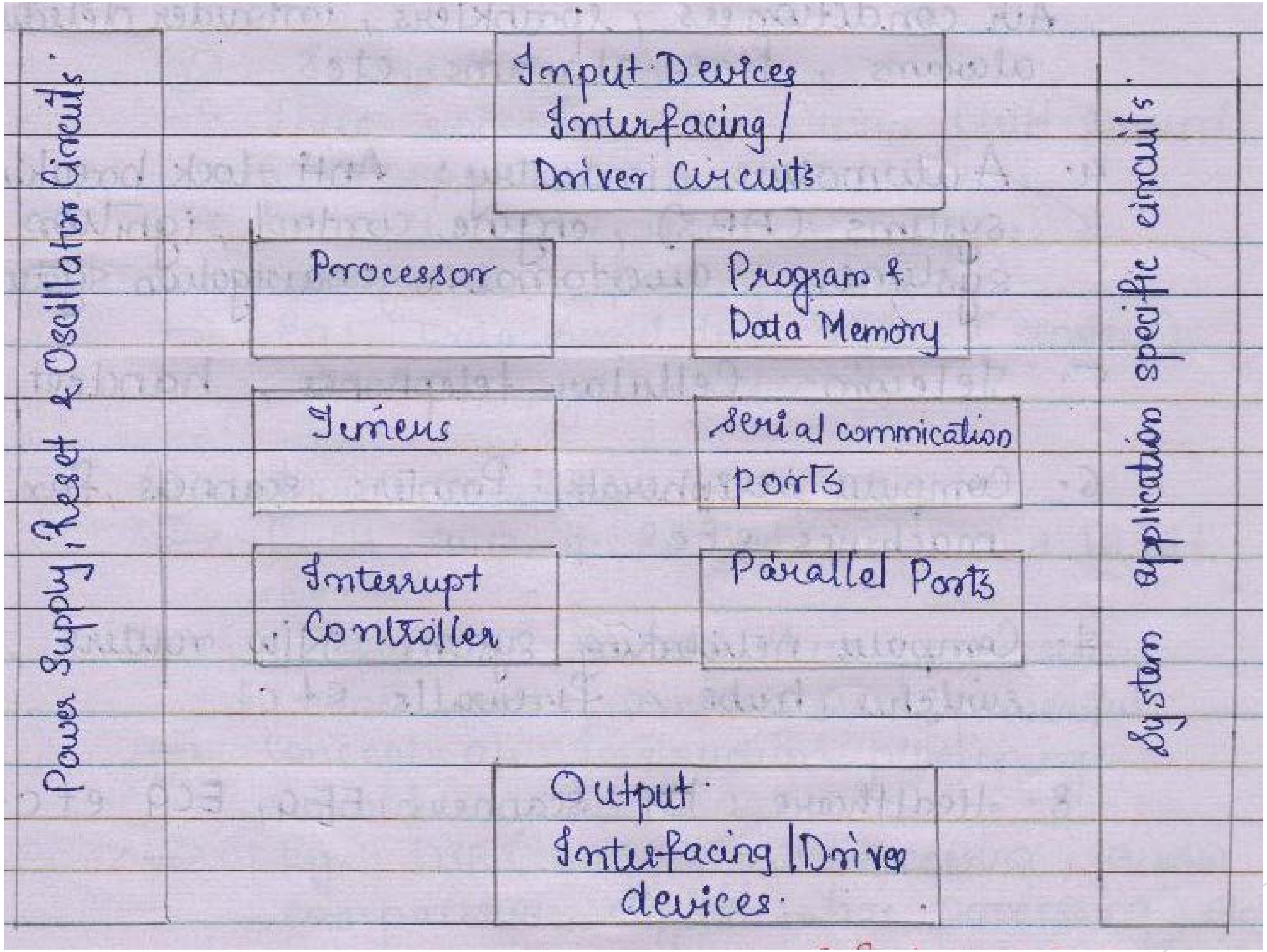
What is an Embedded System?

- *Combination of both hardware and software*
- *Designed to perform specific function*
- *A system that has embedded software and hardware, that makes it a dedicated system for a specific application or a part of application or product is called Embedded System.*
- *It is also defined as a combination of computer hardware with the software to control, monitor, communicate or do multiple operations.*
- *In architectural language it consist of a processor, associated peripherals and software used for specific purpose.*
- *Examples are Digital Camera, Washing Machine, DVD Player, Television, etc.*

General Purpose System v/s Embedded System

General Purpose Computing System	Embedded System
A system which is a combination of a generic hardware and a general purpose operating system for executing variety of applications.	*A system which is a combination of special purpose hardware and embedded OS for executing a specific set of application.*
Contains a general purpose operating system.	*May or may not contain an OS for functioning.*
Applications are alterable by end user.	*The firmware of embedded system is pre-programmed and it is non-alterable by the end user.*
Performance is the key deciding factor in selection of system.	*Application specific requirements (like power requirements, performance, memory usage, etc.) are key deciding factor*
Need not to be deterministic in execution behavior.	*Execution behavior is deterministic for certain types of embedded system like 'Hard Real Time' system.*
Example: computer, laptop	*Example: DVD, mobile phones, printers.*

Embedded System Architecture



Characteristics of Embedded System

- ***Single functioned***
- ***Reliability***
- ***Cost effectiveness***
- ***Low power consumption***
- ***Fast execution time***
- ***Efficient use of memory***
- ***Processing power***

Major Application areas of Embedded System

- ***Consumer electronics:*** *Camcorders, Cameras, etc.*
- ***Household appliances:*** *TV, DVD players, washing machine, refrigerator, oven, etc.*
- ***Home automation & Security system:*** *Air conditioners, sprinklers, intruder detection alarms, fire alarms, etc.*
- ***Automotive industry:*** *Anti-lock breaking system (ABS), engine control, ignition systems, automatic navigation system, etc.*
- ***Telecom:*** *Cellular telephones, handset, etc.*
- ***Computer peripherals:*** *printers, scanners, fax, machines, etc.*
- ***Healthcare:*** *scanners, EEG, ECG, etc.*
- ***Measurement & Instrumentation:*** *Digital multi meters, digital CRO's, etc.*
- ***Banking& Retail:*** *Automatic Teller Machine (ATM), currency counter, etc.*
- ***Card Readers:*** *barcode, smart card readers, handheld devices, etc.*

Categorization of Embedded System

- ***Based on Generation:***
  - ***First Generation:***
    - *These embedded system were built around 4-bit and 8-bit microcontrollers.*
    - *Examples Digital Telephone Keypads, Stepper motor control units.*
  - ***Second Generation:***
    - *These embedded system were built around 8-bit and 16-bit microcontrollers.*
    - *Instruction set is more complex and powerful than first generation.*
    - *Examples Data Acquisition System (sensors), BCADA systems, etc.*
  - ***Third Generation:***
    - *These embedded system were built around 16-bit and 32-bit microcontrollers.*
    - *Instruction set of these processors became more complex and powerful.*
    - *Concept of instruction pipelining evolved.*
    - *Examples DSP (Digital Signal Processor), etc.*
  - ***Fourth Generation:***
    - *The advent of SoC, Multicore processor, reconfigurable processor came into market.*
    - *These system made use of high performance real time embedded OS for their functioning.*
    - *Example Smart Phones, Mobile Internet Devices, etc.*

Categorization of Embedded System (Contd.)

- ***Based on Complexity & Performance:***
  - ***Small Scale Embedded System:***
    - *These include small processing elements, which have minimal input and output and have simple program which runs on the processor.*
    - *Performance requirements are not time critical.*
    - *Low Cost*
    - *8-16 bit Microprocessor/Microcontroller.*
    - *May or may not contain OS.*
    - *Example Electronic toy.*
  - ***Medium Scale Embedded System:***
    - *They are slightly complex in hardware and software.*
    - *Medium Performance*
    - *Low Cost*
    - *16-32 bit Microprocessor/Microcontroller.*
    - *Contains Embedded OS.*
    - *Example ATM*

Categorization of Embedded System (Contd.)

- ***Large Scale Embedded System***
  - *Complex System.*
  - *High Performance.*
  - *Low Cost*
  - *16-32 bit Microprocessor/Microcontroller.*
  - *Employed in mission critical application*
  - *Example Smart phones, multimedia systems, etc.*

Categorization of Embedded System (Contd.)

- ***Based on deterministic behavior:***
  - ***Hard Real Time Embedded System***
    - *Must operate within stringent confined deadline*
    - *Example anti-lock breaks, air craft control system, etc.*
  - ***Soft Real Time Embedded System***
    - *Missing deadline is tolerable, but quality degrades.*
    - *Example online video, etc.*

Categorization of Embedded System (Contd.)

- ***Based on Triggers:***
  - *Event based Trigger*
  - *Time based Trigger*

RISC v/s CISC

RISC	CISC
Instruction set with 70 to 120 instructions.	*Instruction set with 120 to 350 instructions.*
Simple instruction format	*Variable instruction format*
Large set of CPU registers	*Small set of general purpose registers*
Very few addressing modes	*A large number of addressing modes*
Simple pipelining	*Complex pipelining*
Supports on chip cache memory	*Seldom supports on chip cache memory*
Easy to construct superscalar processor using RISC architecture	*Difficult to construct superscalar processor based on CISC architecture*

8051 MICROCONTROLLER

EXPLAINED IN DEPTH

Introduction

- *A decade back the process and control operations were totally implemented by the Microprocessors only.*
- *But now a days the situation is totally changed and it is occupied by the new devices called Microcontroller.*
- *The development is so drastic that we can’t find any electronic gadget without the use of a microcontroller.*
- *This microcontroller changed the embedded system design so simple and advanced that the embedded market has become one of the most sought after for not only entrepreneurs but for design engineers also.*

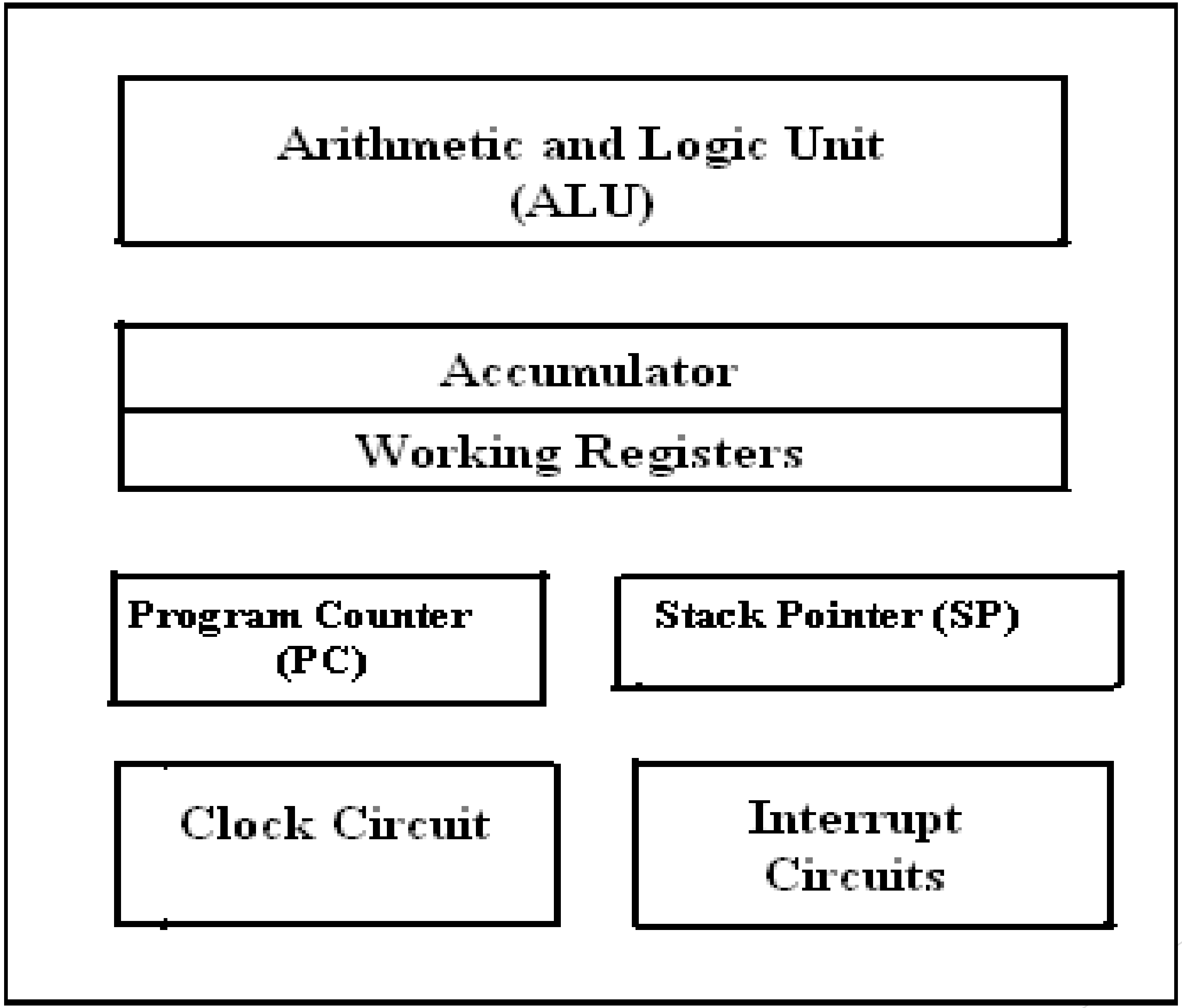
Necessary tools for a Microprocessor/controller:

- ***CPU: Central Processing Unit***
- ***I/O: Input /Output***
- ***Bus: Address bus & Data bus***
- ***Memory: RAM & ROM***
- ***Timer***
- ***Interrupt***
- ***Serial Port***
- ***Parallel Port***

What is Microprocessor?

- *A CPU built into single VSLI chip is called a microprocessor.*
- *The microprocessor contains : Arithmetic and logic unit (ALU), Accumulator Instruction register, Program counter (PC), clock circuit (internal or external), reset circuit (internal or external) registers*
- *But the microprocessor has no on chip I/O Ports, Timers , Memory etc.*
- *For example, Intel 8085 is an 8-bit microprocessor and Intel 8086/8088 a 16-bit microprocessor.*

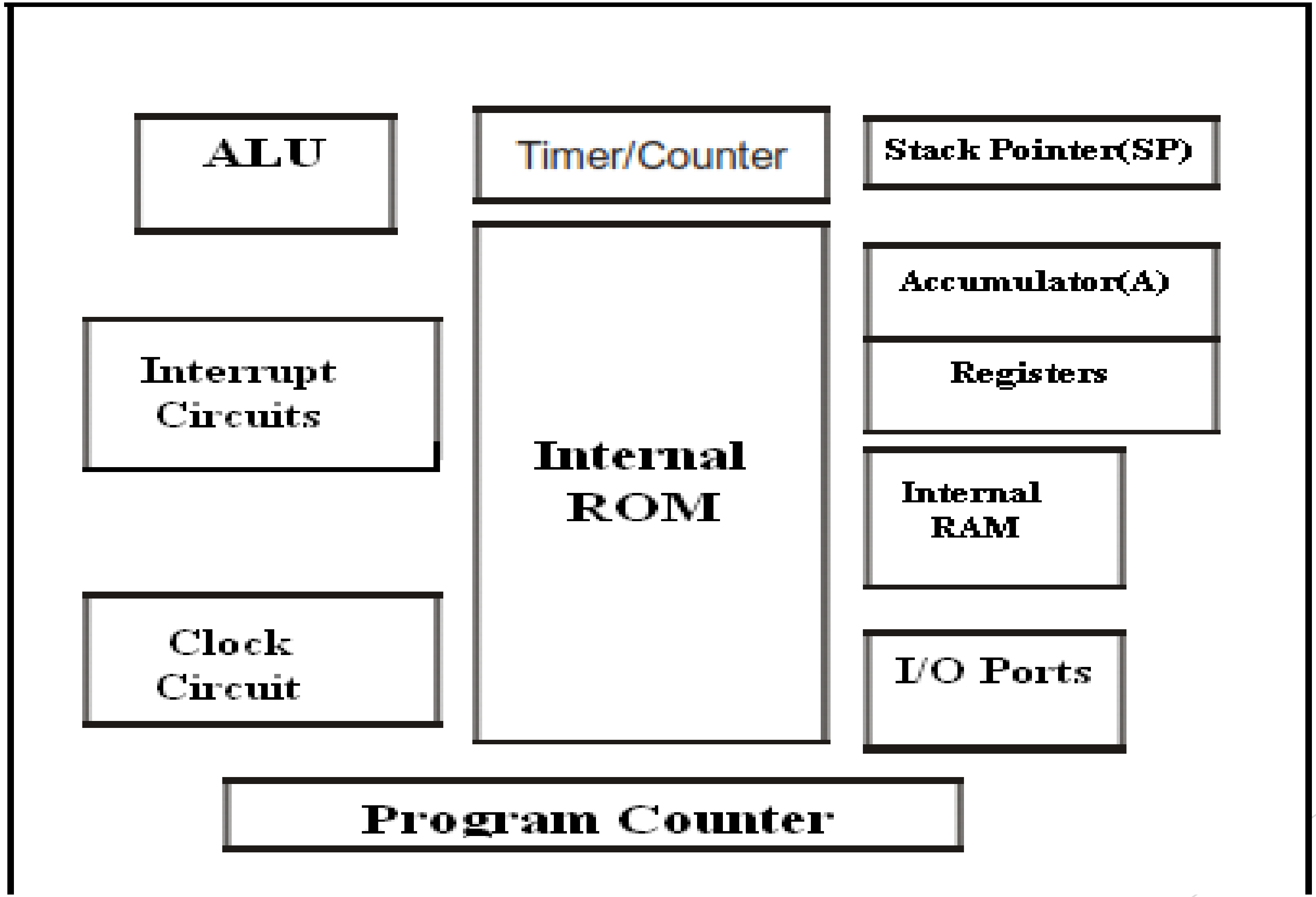
Basic Microprocessor Block Diagram



What is Microcontroller?

- *A microcontroller is a highly integrated single chip, which consists of*
  - *on chip CPU (Central Processing Unit),*
  - *RAM (Random Access Memory),*
  - *EPROM/PROM/ROM (Erasable Programmable Read Only Memory), I/O (input/output) – serial and parallel,*
  - *timers,*
  - *interrupt controller.*
- *For example, Intel 8051 is 8-bit microcontroller and Intel 8096 is 16-bit microcontroller.*

Basic Microcontroller Block Diagram



Microprocessor v/s Microcontroller

Microprocessor	Microcontroller
Microprocessor use CISC architecture	Microcontroller use RISC and Harvard architecture
Microprocessor has ROM, RAM, secondary storage memory, I/O peripherals, etc. placed on board and connected through buses	In Microcontroller all three peripherals are combined in a single integrated circuit
Microprocessor are slow in terms of overall performance and application execution.	Microcontroller are fast in terms of overall performance and application execution.
Designing of Microprocessor takes more time	Designing of Microcontroller takes less time
Microprocessor is less secured as compared to Microcontroller	Microcontroller is more secured.
Microprocessor are expensive	Microcontroller are cheap as compared Microprocessor

What is 8051 Microcontroller ?

- ***Single chip microcontroller (µC)***
- ***Developed by Intel in 1980 for use in embedded systems***
- ***First microcontroller of the MCS-51family***

Inside the Block Diagram of 8051

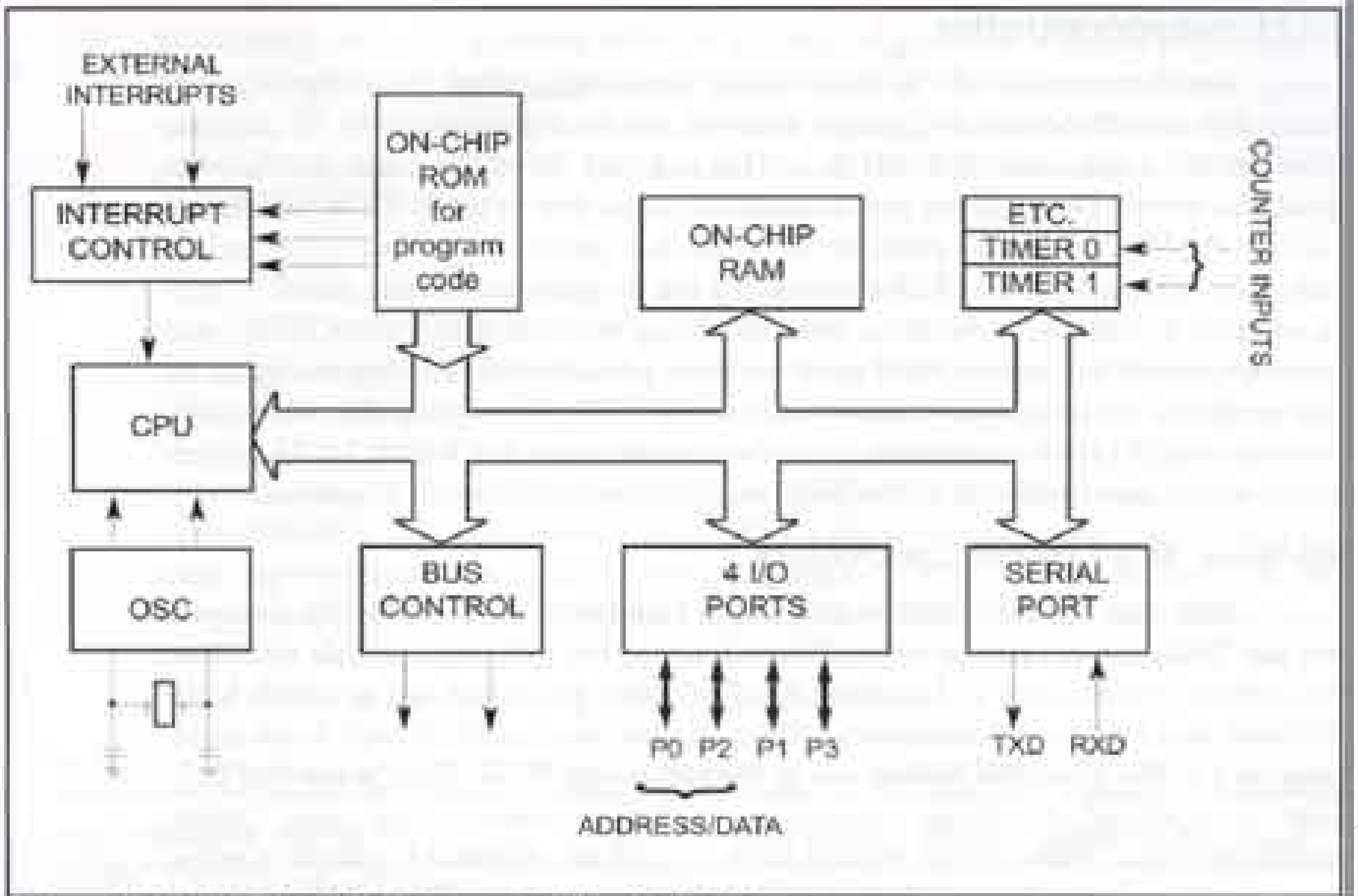


Figure 1-2. Inside the 8051 Microcontroller Block Diagram

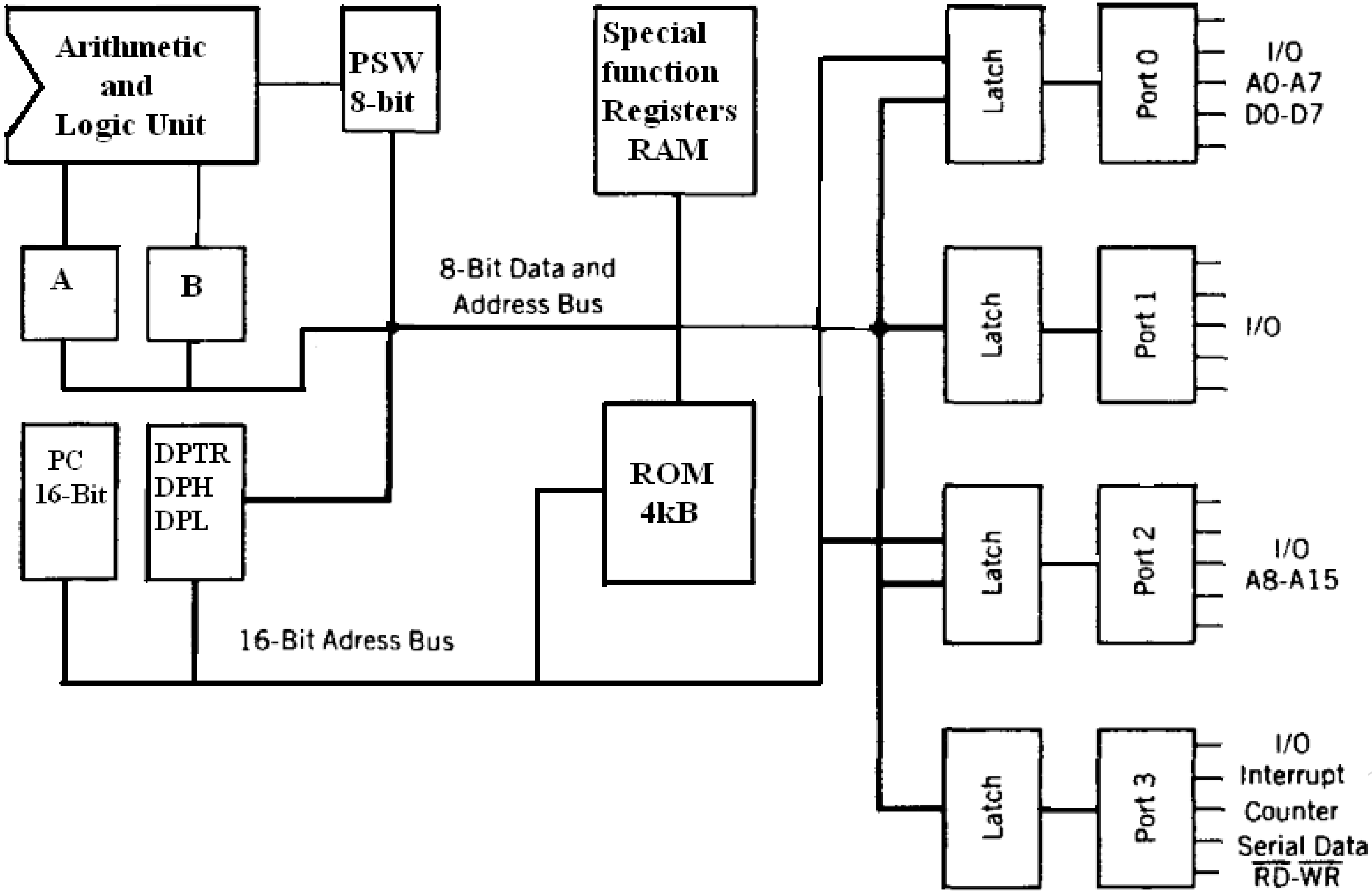
Salient Features of 8051 Microcontroller:

- ***8-bit CPU (CPU can work on only 8 bits of data at a time)***
- ***Two 16 bit timers/counters***
- ***128 bytes of RAM***
- ***4K bytes of on-chip ROM***
- ***One serial port***
- ***8-bit data bus***
- ***16-bit address bus***
- ***One 16-bit program counter and One 16-bit DPTR (data pointer)***
- ***One 8-bit stack pointer***
- ***32 general purpose registers each of 8 bits***
- ***Four I/O ports, each 8 bits wide***
- ***6 interrupt sources***
- ***On-chip clock oscillator***
- ***Control registers: TCON, TMOD, SCON, PCON, IP and IE.***

Architecture Of 8051 Microcontroller

- *The architecture of the 8051 microcontroller can be understood from the following Fig.*
- *It has Harvard architecture with RISC (Reduced Instruction Set Computer) concept.*
- *It consists of :*
  - *8-bit ALU,*
  - *one 8-bit PSW (Program Status Register),*
  - *A and B registers ,*
  - *one 16-bit Program counter ,*
  - *one 16-bit Data pointer register (DPTR),*
  - *128 bytes of RAM*
  - *4kB of ROM and*
  - *four parallel I/O ports each of 8-bit width.*

Functional Block Diagram of 8051 Microcontroller



Architecture Of 8051 Microcontroller

- ***Arithmetic and logical unit (ALU) :***
  - *8051 has 8-bit ALU which can perform all the 8-bit arithmetic and logical operations in one machine cycle.*
  - *The ALU is associated with two registers A & B.*

- ***A & B Registers :***
  - *The A and B registers are special function registers .*
  - *Hold the results of many arithmetic and logical operations of 8051.*
  - *The A register is also called the Accumulator.*
  - *It is used as a general register to accumulate the results of a large number of instructions.*
  - *By default it is used for all mathematical operations and also data transfer operations between CPU and any external memory.*
  - *The B register is mainly used for multiplication and division operations along with A register.*
- *MUL AB* *DIV AB.*

Architecture Of 8051 Microcontroller (Contd.)

- *Program Counter(PC) :*
  - *8051 has a 16-bit program counter .*
  - *Points to the address of the next instruction to be executed.*
  - *After execution of one instruction the program counter is incremented to point to the address of the next instruction to be executed.*
  - *It is the contents of the PC that are placed on the address bus to find and fetch the desired instruction.*
  - *Since the PC is 16-bit width ,8051 can access program addresses from 0000H to FFFFH ,a total of 6kB of code.*

Architecture Of 8051 Microcontroller (Contd.)

- ***STACK in 8051 Microcontroller :***
  - *The stack is a part of RAM used by the CPU to store information temporarily.*
  - *This information may be either data or an address.*
  - *The CPU needs this storage area as there are only limited number of registers.*
  - *The register used to access the stack is called the Stack pointer.*
  - *There are two important instructions to handle this stack: **PUSH** and **POP**.*
  - *The loading of data from CPU registers to the stack is done by **PUSH** and the loading of the contents of the stack back into a CPU register is done by **POP**.*

Architecture Of 8051 Microcontroller (Contd.)

- ***Stack Pointer Register (SP) :***
  - *It is an 8-bit register*
  - *Stores the address of the stack top. i.e. the Stack Pointer is used to indicate where the next value to be removed from the stack should be taken from.*
  - *When a value is pushed onto the stack, the 8051 first increments the value of SP and then stores the value at the resulting memory location.*
  - *Similarly when a value is popped off the stack, the 8051 returns the value from the memory location indicated by SP, and then decrements the value of SP.*
  - *Since the SP is only 8-bit wide it is incremented or decremented by two .*
  - *SP is modified directly by the 8051 by six instructions: PUSH, POP, ACALL, LCALL, RET, and RETI.*
  - *It is also used intrinsically whenever an interrupt is triggered.*

Architecture Of 8051 Microcontroller (Contd.)

- ***Data Pointer Register (DPTR) :***
  - *It is a 16-bit register which is the only user-accessible.*
  - *Used to point to data.*
  - *Used by a number of commands which allow the 8051 to access external memory.*
  - *When the 8051 accesses external memory it will access external memory at the address indicated by DPTR.*
  - *This DPTR can also be used as two 8-registers : **DPH** and **DPL**.*

Architecture Of 8051 Microcontroller (Contd.)

- ***Program Status Register (PSW) :***
  - *The 8051 has a 8-bit PSW register*
  - *Also known as Flag register.*
  - *In the 8-bit register only 6-bits are used by 8051.The two unused bits are user definable bits.*
  - *In the 6-bits four of them are conditional flags .They are:*
    - *Carry –CY,*
    - *Auxiliary Carry-AC,*
    - *Parity-P, and*
    - *Overflow-OV .*
  - *These flag bits indicate some conditions that resulted after an instruction was executed.*

PSW register

D7 D0

CY	AC	FO	RS1	RS0	OV	-	P

Architecture Of 8051 Microcontroller (Contd.)

- The bits PSW3 and PSW4 are denoted as RS0 and RS1 and these bits are used the select the bank registers of the RAM location. The meaning of various bits of PSW register is shown below.

CY	***PSW.7***	***Carry Flag***
AC	*PSW.6*	*Auxiliary Carry Flag*
FO	*PSW.5*	*Flag 0 available for general purpose*
RS1	*PSW.4*	*Register Bank select bit 1*
RS0	*PSW.3*	*Register bank select bit 0*
OV	*PSW.2*	*Overflow flag*
---	*PSW.1*	*User definable flag*
P	*PSW.0*	*Parity Flag*

Architecture Of 8051 Microcontroller (Contd.)

RS0	*RS1*	*Register Bank*	*Address*
0	*0*	*0*	*00H - 07H*
0	*1*	*1*	*08H – 0FH*
1	*0*	*2*	*10H – 17H*
1	*1*	*3*	*18H - 1FH*

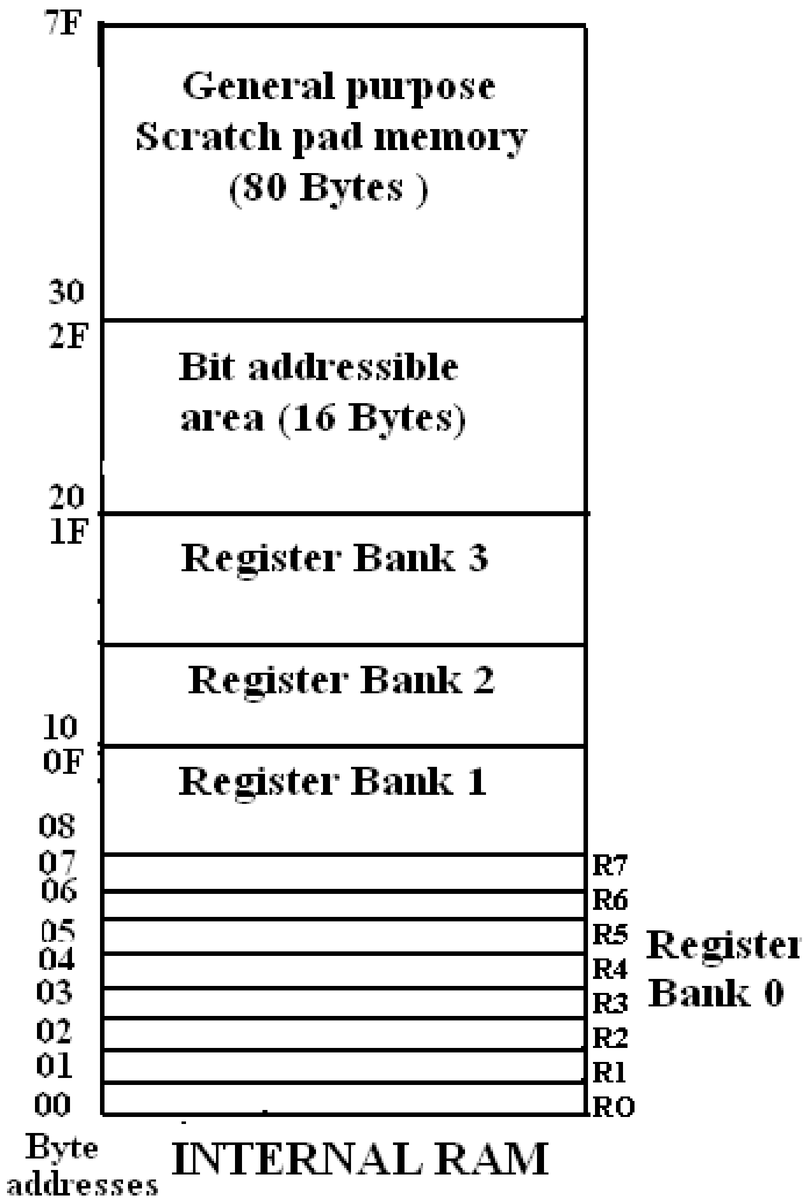
Memory Organization in 8051 Microcontroller

- *The 8051 microcontroller has 128 bytes of Internal RAM and 4kB of on chip ROM .*
- *The RAM is also known as Data memory and the ROM is known as program memory (Code memory).*
- *This Code memory   holds the actual 8051 program that is to be executed.*
- *In 8051 this memory is limited to 64K .Code memory may be found on-chip, as ROM or EPROM.*
- *It may also be stored completely off-chip in an external ROM or, more commonly, an external EPROM.*
- *The 8051 has only 128 bytes of Internal RAM but it supports 64kB of external RAM.*

Internal RAM OF 8051

- *This Internal RAM is found on-chip on the 8051 .*
- *So it is the fastest RAM available, and it is also the most flexible in terms of reading, writing, and modifying it's contents.*
- *Internal RAM is volatile, so when the 8051 is reset this memory is cleared.*
- *The 128 bytes of internal RAM is organized as below.*
  - *Four register banks (Bank0,Bank1, Bank2 and Bank3) each of 8-bits (total 32 bytes).*
  - *The default bank register is Bank0.*
  - *The remaining Banks are selected with the help of RS0 and RS1 bits of PSW Register.*
  - *16 bytes of bit addressable area   and*
  - *80 bytes of general purpose area (Scratch pad memory) as shown in the diagram below.*
  - *This area is also utilized by the microcontroller as a storage area for the operating stack.*

Internal RAM OF 8051



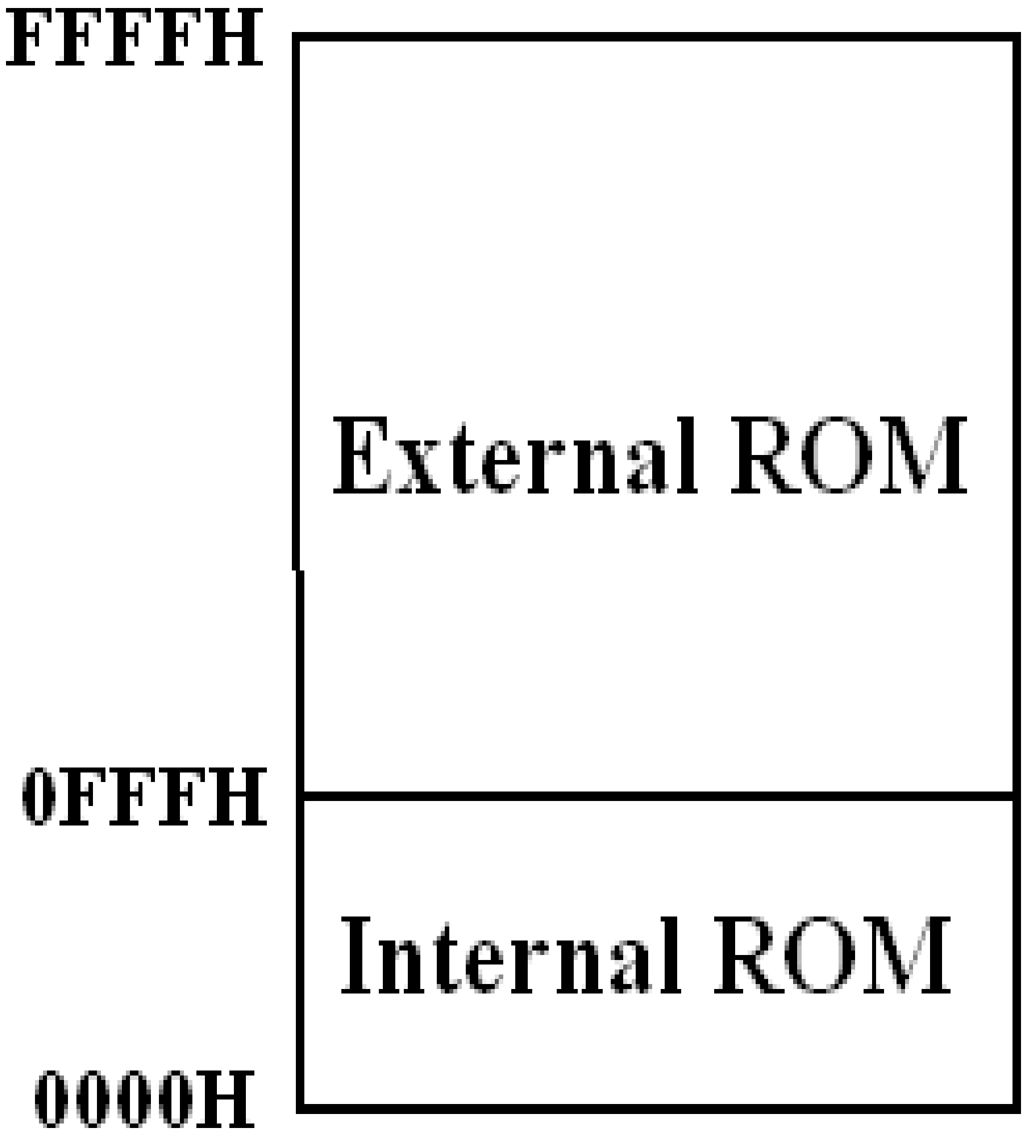
Internal RAM OF 8051

- ***Register banks:***
  - *The 32 bytes of RAM from address 00 H to 1FH are used as working registers organized as four banks of eight registers each.*
  - *The registers are named as R0-R7 .*
  - *Each register can be addressed by its name or by its RAM address.*
  - *For example MOV A, R7 or MOV R7,#05H*
- ***Bit Addressable RAM:***
  - *It occupies 16 bytes of RAM between 20h to 2Fh.*
  - *Addressable bits are useful in case of binary event (switch on , light off etc.) where only one bit is needed.*
- ***General purpose RAM:***
  - *It occupies 80 bytes of RAM between 30h to 7Fh.*
  - *It is used for general purpose data storage.*
  - *System stack uses this memory.*

Internal ROM (On –chip ROM)

- *The 8051 microcontroller has 4kB of on chip ROM but it can be extended up to 64kB.*
- *The CODE segment is accessed using the program counter (PC) for opcode fetches and by DPTR for data.*
- *The external ROM is accessed when the EA (active low) pin is connected to ground or the contents of program counter exceeds 0FFFH.*
- *When the Internal ROM address is exceeded the 8051 automatically fetches the code bytes from the external program memory.*

Internal ROM (On –chip ROM)



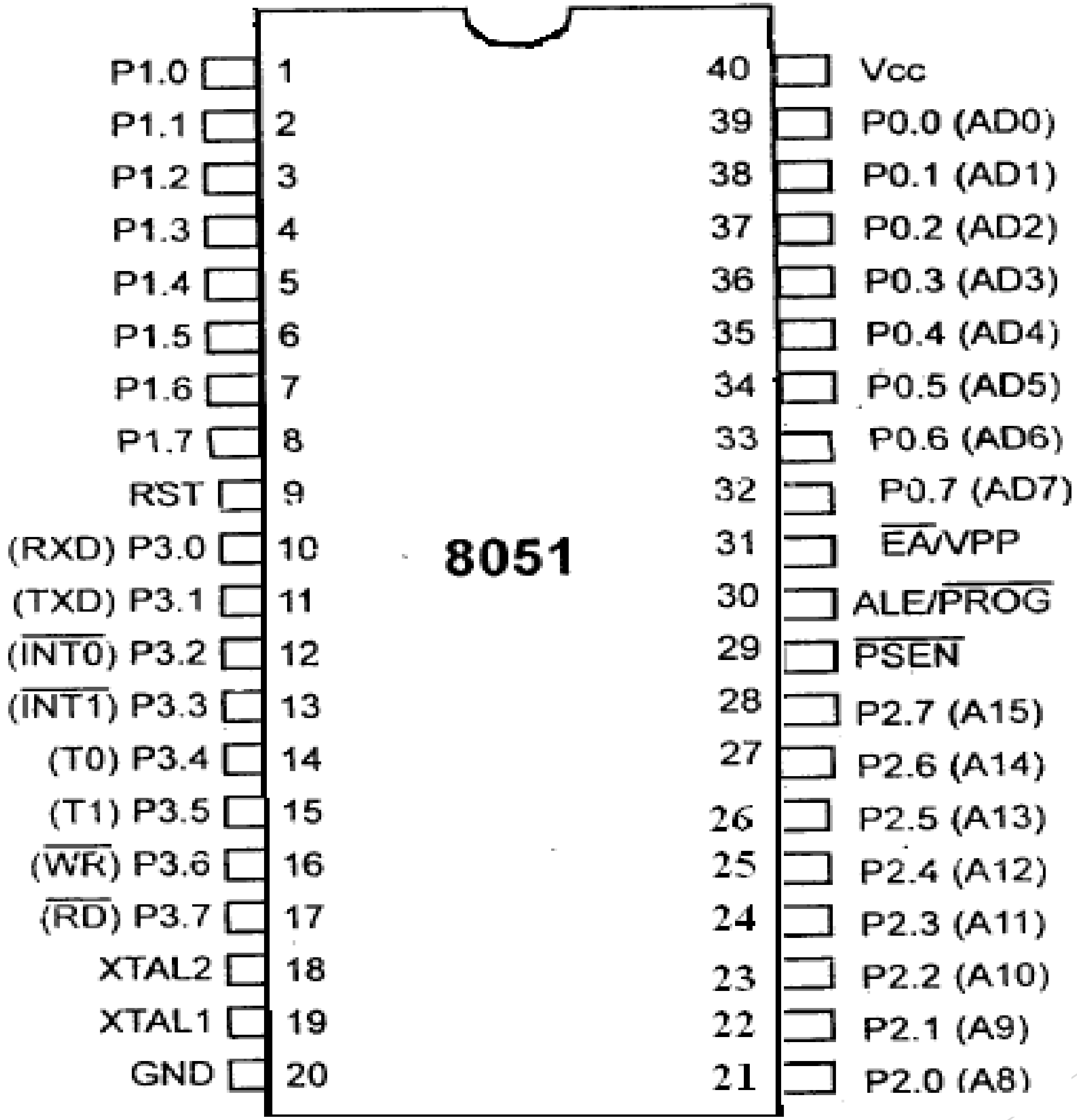
SPECIAL FUNCTION REGISTERS (SFRs)

- *In 8051 microcontroller there certain registers which uses the RAM addresses from 80H to FFH and they are meant for certain specific operations .*
- *These registers are called Special function registers (SFRs).*
- *Some of these registers are bit addressable also.*
- *The list of SFRs and their functional names are given below.*
- *In these SFRs some of them are related to I/O ports (P0,P1,P2 and P3) and some of them are meant for control operations (TCON,SCON, PCON..) and remaining are the auxiliary SFRs.*

Pin Description of the 8051

- *The 8051 microcontroller is available as a 40 pin DIP chip.*
- *Works at +5 volts DC*
- *Among the 40 pins , a total of 32 pins are allotted for the four parallel ports P0,P1,P2 and P3 i.e. each port occupies 8-pins.*
- *The remaining pins are VCC, GND, XTAL1, XTAL2, RST, EA ,PSEN,ALE.*

Pin Diagram of 8051



Pin Description of 8051

- ***Pin 40:***
  - *Named as Vcc is the main power source usually +5V DC.*
- ***Pin 32-39 (Port 0):***
  - *Known as Port 0 – In addition to serving as I/O port, lower order address and data bus signals are multiplied with its port. This is a bi-directional I/O port and external pull up registers are required to function this port as I/O.*
- ***Pin 31:***
  - *External Access input is used to enable or disable the external memory interfacing. If there is no external memory requirement, this pin is pulled high by connecting it to Vcc.*
- ***Pin 30:***
  - *ALE aka Address Latch Enable is used to DE multiplex the address-data signals of port 0. Two ALE pulses are available for each machine cycle.*
- ***Pin 29:***
  - *PSEN or Program Store ENable is used to read signal from external program memory.*

Pin Description of 8051

- ***<u>Pin 28-21 (Port 2):</u>***
  - Known as Port 2 - In addition to serving as I/O port, higher order address and data bus signals are multiplexed with this quasi bi-directional port.
- ***<u>Pin 20:</u>***
  - This pin represents ground connection.
- ***<u>Pin 18 and 19:</u>***
  - Used for interfacing an external crystal to provide system clock.
- ***<u>Pin 17-10 (Port 3):</u>***
  - Known as Port 3 – This port also serves as some other functions like interrupts, timer inputs, control signals for external memory interfacing RD and WR, serial communication signals RxD & TxD, etc.
- ***<u>Pin 9:</u>***
  - RESET pin is used to set the 8051 microcontroller to its initial values, while the microcontroller is working or at the initial start of the application.

Pin Description of 8051

- ***Pin 1-8:***
  - Known as Port 1. Unlike other ports, this port does not serve any other function. Port 1 is an internally pulled up, quasi bidirectional I/O port.

Interrupt Structure in 8051

- *An interrupt is an external or internal event that disturbs the microcontroller to inform it that a device needs its service.*
- *The program which is associated with the interrupt is called the interrupt service routine (ISR) or interrupt handler.*
- *Upon receiving the interrupt signal, the Microcontroller finish current instruction and saves the PC on stack.*
- *Jumps to a fixed location in memory depending on type of interrupt.*
- *Starts to execute the interrupt service routine until RETI (return from interrupt).*
- *Upon executing the RETI the microcontroller returns to the place where it was interrupted.*
- *Get pop PC from stack.*

Interrupt Structure in 8051

- ***Classification of Interrupts:***
  - *Interrupts can be classified into two types:*
    - *External Interrupts*
    - *Internal Interrupts*
- ***Interrupt mask ability:***
  - *Interrupt requests can be classified into 2 types:*
    - ***Mask able Interrupts-*** *Microcontrollers have the option to disable the interrupts. These interrupts are called Mask able Interrupt.*
    - ***Non-Mask able Interrupts****-Interrupts that can't be disabled by Microcontroller is called Non-Mask able Interrupts*
- ***The 8051 microcontroller has FIVE interrupts in addition to Reset. They are:***
  - *Timer 0 overflow Interrupt*
  - *Timer 1 overflow Interrupt*
  - *External Interrupt 0(INT0)*
  - *External Interrupt 1(INT1)*
  - *Serial Port events (buffer full, buffer empty, etc.) Interrupt*

Interrupt Structure in 8051

- *Each interrupt has a specific place in code memory where program execution (interrupt service routine) begins.*
- *This is shown by interrupt vector table.*
- *Interrupt Vector Table:*
  - *External Interrupt 0: 0003 H*
  - *Timer 0 overflow: 000B H*
  - *External Interrupt 1: 0013 H*
  - *Timer 1 overflow: 001B H*
  - *Serial Interrupt : 0023 H*
- *Upon reset all Interrupts are disabled & do not respond to the Microcontroller.*
- *These interrupts must be enabled by software in order for the Microcontroller to respond to them. This is done by an 8-bit register called Interrupt Enable Register (IE).*

Interrupt Enable Register

EA	---	---	ES	ET1	EX1	ET0	EX0

- *EA : Global enable/disable. To enable the interrupts this bit must be set High.*
- *--- : Undefined-reserved for future use.*
- *--- : Enable/disable Timer 2 overflow interrupt.*
- *ES : Enable/disable Serial port interrupt.*
- *ET1 : Enable/disable Timer 1 overflow interrupt.*
- *EX1 : Enable/disable External interrupt 1.*
- *ET0 : Enable /disable Timer 0 overflow interrupt.*
- *EX0 : Enable/disable External interrupt 0.*

Interrupt Enable Register

- *Upon reset the interrupts have the following priority (Top to down).*
- *The interrupt with the highest PRIORITY gets serviced first.*
- *External interrupt 0 (INT0)*
- *Timer interrupt 0 (TF0)*
- *External interrupt 1 (INT1)*
- *Timer interrupt1 (TF1)*
- *Serial communication (RI+TI)*
- *Priority can also be set to “high” or “low” by 8-bit Interrupt Priority register.*

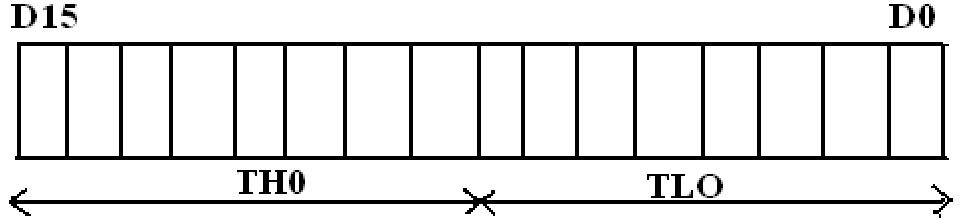
Interrupt Priority Register

- *IP.7: reserved*
- *IP.6: reserved*
- *IP.5: reserved*
- *IP.4: Serial port interrupt priority bit*
- *IP.3: Timer 1 interrupt priority bit*
- *IP.2: External interrupt 1 priority bit*
- *IP.1: Timer 0 interrupt priority bit*
- *IP.0: External interrupt 0 priority bit*
- *Interrupt priority register as shown below:*

---	---	---	PS	PT1	PX1	PT0	PX0

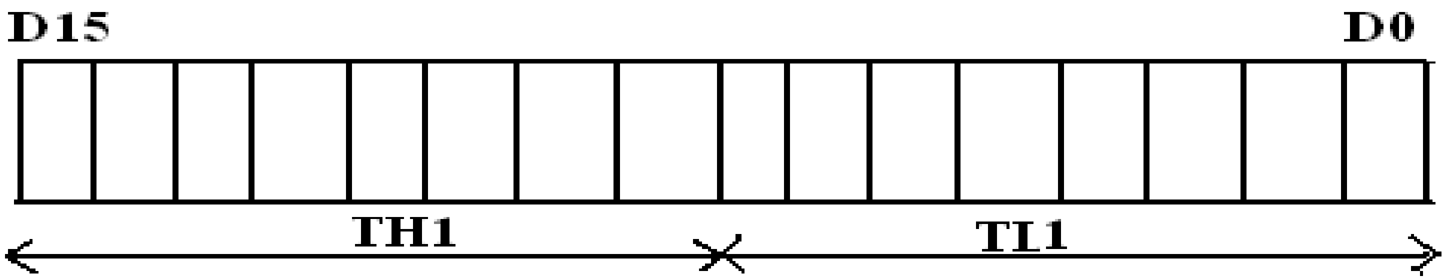
TIMERS/COUNTERS in 8051 Microcontrollers

- *The 8051 microcontroller has two 16-bit timers Timer 0 (T0) and Timer 1(T1).*
- *Timers can be used to generate accurate time delays whereas counters are used to count events happening outside the microcontroller.*
- *These timers are accessed as two 8-bit registers TLO, THO & TL1 ,TH1 because the 8051 microcontroller has 8-bit architecture.*
- ***<u>TIMER 0:</u>***
  - *The Timer 0 is a 16-bit register and can be treated as two 8-bit registers (TL0 & TH0).*
  - *These registers can be accessed similar to any other registers like A,B or R1,R2,R3 etc.*
  - *For example the instruction MOV TL0,#07 moves the value 07 into lower byte of Timer0.*
  - *Similarly MOV R5,TH0 saves the contents of TH0 in the R5 register.*



TIMERS/COUNTERS in 8051 Microcontrollers

- ***TIMER 1:***
  - *The Timer 1 is also a 16-bit register and can be treated as two 8-bit registers (TL1 & TH1).*
  - *These registers can be accessed similar to any other registers like A,B or R1, R2, R3 etc.*
  - *For example the instruction MOV TL1,#05 moves the value 05 into lower byte of Timer1.*
  - *Similarly MOV R0,TH1 saves the contents of TH1 in the R0 register.*



TMOD Register:

- *The various operating modes of both the timers T0 and T1 are set by an 8-bit register called TMOD register.*
- *In this TMOD register the lower 4-bits are meant for Timer 0 and the higher 4-bits are meant for Timer1.*
  - ***GATE:***
    - *This bit is used to start/stop the timers by hardware .*
    - *When GATE= 1 ,the timers can be started/stopped by the external sources.*
    - *When GATE= 0, the timers can be started/stopped by software instructions like SETB TR0 or SETB TR1*
  - ***C/T (clock/Timer):***
    - *This bit decides whether the timer is used as delay generator or event counter.*
    - *When C/T = 0 ,the Timer is used as delay generator and if C/T=1 the timer is used as an event counter.*
    - *The clock source for the time delay is the crystal frequency of 8051.*

TMOD Register

- ***M1,M0 (Mode):***
  - *These two bits are the timer mode bits.*
  - *The timers of the 8051 can be configured in three modes-Mode 0, Mode 1, Mode 2 and Mode 3.*

D7							D0
GATE	$C/\overline{T}$	M1	M0	GATE	$C/\overline{T}$	M1	M0
Timer1				Timer 0			

TMOD Register

- *The selection and operation of the modes is shown below.*

Sr.no.	M0	M1	Mode	Operation
1	0	0	0	13 bit timer TH -> 08 bit TL -> 05 bit
2	0	1	1	16 bit timer TH -> 08 bit TL -> 08 bit
3	1	0	2	08 bit timer TH = TL -> 8 bit
4	1	1	3	Split Timer Mode

TCON Register

- *TCON stands for Timer control register:*
- *Bit addressable register*
- ***TR (run control bit):***
  - *TR0 for Timer/counter 0*
  - *TR1 for Timer/counter 1.*
  - *TRx is set by programmer to turn timer/counter on/off.*
  - *TRx=0: off (stop)*
  - *TRx=1: on (start)*

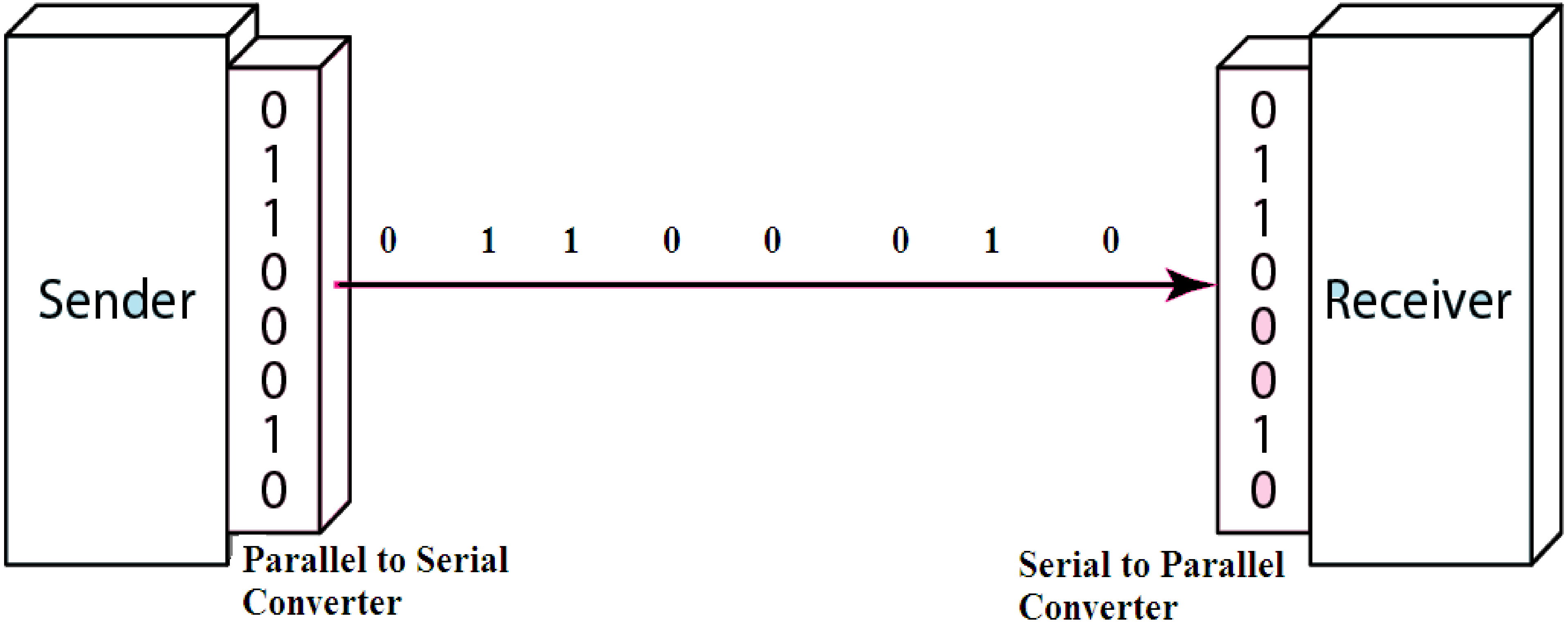
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

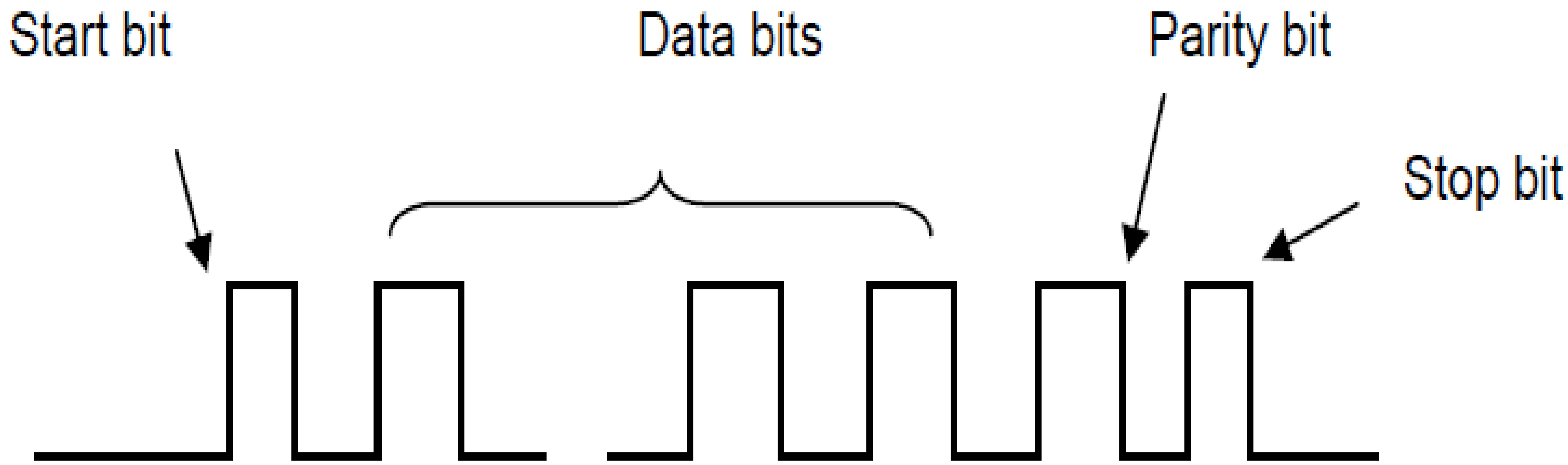
TCON Register

- ***TF (timer flag):***
  - *TF0 for timer/counter 0*
  - *TF1 for timer/counter 1.*
  - *TFx is like a carry.*
  - *Originally, TFx=0. When TH-TL roll over to 0000 from FFFFH, the TFx is set to 1*
  - *TFx=0 : not reach*
  - *TFx=1: reach*
  - *If we enable interrupt, TFx=1 will trigger ISR.*
- ***IE1, IE0:***
  - *Edge flag for external interrupts 1 and 0.*
  - *Set by interrupt edge, cleared when interrupt is processed.*
- ***IT1, IT0:***
  - *Type bit for external interrupts.*
  - *Set for falling edge interrupts, reset for 0 level interrupts*

Basics of Serial communication

- *Data transfer between two electronic devices (Ex Between a computer and microcontroller or a peripheral device) is generally done in two ways:*
  - *Serial data Transfer*
  - *Parallel data Transfer*
- *Serial communication uses only one or two data lines to transfer data and is generally used for long distance communication. In serial communication the data is sent as one bit at a time in a timed sequence on a single wire.*
- *Serial Communication takes place in two methods:*
  - *Asynchronous data Transfer*
  - *Synchronous data Transfer.*
- *Asynchronous data transfer allows data to be transmitted without the sender having to send a clock signal to the receiver. Instead, special bits will be added to each word in order to synchronize the sending and receiving of the data. When a word is given to the UART for Asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted. The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter.*


Sender
0
1
1
0
0
0
1
0
0 1 1 0 0 0 1 0
Receiver
0
1
1
0
0
0
1
0
Parallel to Serial Converter
Serial to Parallel Converter


Start bit
Data bits
Parity bit
Stop bit

Basics of Serial communication

- *After the Start Bit, the individual bits of the word of data are sent .Here each bit in the word is transmitted for exactly the same amount of time as all of the other bits. When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates. The Parity bit may be used by the receiver to perform simple error checking. Then at least one Stop Bit is sent by the transmitter. If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be corrupted and will report a Framing Error.*
- *Baud rate is a measurement of transmission speed in asynchronous communication ,it represents the number of bits/sec that are actually being sent over the serial link. The Baud count includes the overhead bits Start, Stop and Parity that are generated by the sending UART and removed by the receiving UART.*
- *In the Synchronous data transfer method the receiver knows when to "read" the next bit coming from the sender. This is achieved by sharing a clock between sender and receiver. In most forms of serial Synchronous communication, if there is no data available at a given time to transmit, a fill character will be sent instead so that data is always being transmitted. Synchronous communication is usually more efficient because only data bits are transmitted between sender and receiver, however it will be more costly because extra wiring and control circuits are required to share a clock signal between the sender and receiver.*

Basics of Serial communication

- *Devices that use serial cables for their communication are split into two categories:*
  - *DTE (Data Terminal Equipment). Examples of DTE are computers, printers & terminals.*
  - *DCE (Data Communication Equipment). Example of DCE is modems.*
- *Parallel Data Transfer :*
  - *Parallel communication uses multiple wires (bus) running parallel to each other, and can transmit data on all the wires simultaneously. i.e. all the bits of the byte are transmitted at a time. So, speed of the parallel data transfer is extremely high compared to serial data transfer. An 8-bit parallel data transfer is 8-times faster than serial data transfer. Hence with in the computer all data transfer is mainly based on Parallel data transfer. But only limitation is due to the high cost ,this method is limited to only short distance communications.*

Serial communication in 8051

- *The 8051 has two pins for transferring and receiving data by serial communication. These two pins are part of the Port3(P3.0 &P3.1) .*
- *The 8051 has serial communication capability built into it.*
  - *Half-duplex*
  - *Asynchronous mode only.*
- *3 special function registers support serial communication:*
  - *SBUF Register*
  - *SCON Register*
  - *PCON Register*

Serial communication in 8051

- *SBUF Register*
  - *8-bit register*
  - *Two separate SBUF registers:*
    - *One for transmission*
    - *One for reception*
  - *For a byte of data to be transferred via the TxD line, it must be placed in the SBUF.*
  - *SBUF holds the byte of data when it is received by the 8051 RxD line.*
  - *Not bit-addressable*

Serial communication in 8051

- SCON (Serial control) register :
- Serial communication is controlled by an 8-bit register called SCON register, it is a bit addressable register.

SM0	***SM1***	***SM2***	***REN***	***TB8***	***RB8***	***TI***	***RI***

Serial communication in 8051

SM0	SCON.7	Serial port mode selector
SM1	SCON.6	Serial port mode selector
SM2	SCON.5	Used for multiprocessor mode communication (not applicable for 8051).
REN	SCON.4	Receive enable. Set or cleared by making this bit either 1 or 0 for enable /disable reception.
TB8	SCON.3	9th data bit transmitted in modes 2 and 3.
RB8	SCON.2	9th data bit received in modes 2 and 3.it is not used in mode 0 & mode 1.If SM2 = 0 RB8 is the stop bit
TI	SCON.1	Transmit interrupt flag
RI	SCON.0	Receive interrupt flag

Serial communication in 8051

- **SM0 , SM1:**
  - These two bits of SCON register determine the framing of data by specifying the number of bits per character and start bit and stop bits. There are 4 serial modes.

SM0	SM1	
0	0	Serial Mode 0
0	1	Serial Mode 1, 8 bit data, 1 stop bit, 1 start bit
1	0	Serial Mode 2
1	1	Serial Mode 3

Serial communication in 8051

- REN (Receive Enable) also referred as SCON.4. When it is high, it allows the 8051 to receive data on the RxD pin. So to receive and transfer data REN must be set to 1.When REN=0,the receiver is disabled. This is achieved as below

  SETB SCON.4 & CLR SCON.4

- TI (Transmit interrupt) is the D1 bit of SCON register. When 8051 finishes the transfer of 8-bit character, it raises the TI flag to indicate that it is ready to transfer another byte. The TI bit is raised at the beginning of the stop bit.
- RI (Receive interrupt) is the D0 bit of the SCON register. When the 8051 receives data serially ,via RxD, it gets rid of the start and stop bits and places the byte in the SBUF register. Then it raises the RI flag bit to indicate that a byte has been received and should be picked up before it is lost. RI is raised halfway through the stop bit.